Journal of Knowledge Information Technology and Systems

ISSN 1975-7700

http://www.kkits.or.kr

Improvement of Dynamic Web Vulnerability Inspection Method and Procedure by Website Structuring and Calculating Each Page's Action Size

Jae-Ho Lee[1], Sang-Joon Lee[*2]

[1]*Department of Interdisciplinary Program of Information Security, Graduate School Chonnam National University*

[2]*School of Business Administration, Chonnam National University*

ABSTRACT

As the Web evolves, various web vulnerabilities are being discovered. Many companies and organizations are working to eliminate Web vulnerabilities, but they are not shrinking. Due to the nature of web vulnerability checks, dynamic checks are essential, and manual checks are preferred for accurate checking. In the case of a dynamic inspection performed manually, there are various problems such as false negative, missing inspection target and deflection due to inspectors. In this paper, we propose inspection methods and procedures to prevent a false negative, missing inspection target and deflection due to inspectors. In the proposed method, the web site is structured by Information Architecture(IA), and the detailed pages are classified into seven operation functions. The detailed pages are obtained by using the number of parameters, and the size of the entire website is calculated by adding the sizes of the detailed pages. Based on the number of Web vulnerability items to be used for the check, calculate the size of the page that can be checked in one day, and calculate the total inspection schedule. We verified the validity of the proposed method by comparing the number of vulnerabilities detected by the proposed method and the current method, and by analyzing the results of questionnaires for the related field workers. The proposed method can be applied not only to dynamic inspection but also to static inspection.

KEYWORDS: Web application, Web vulnerability, Dynamic inspection, Manual inspection, IA(Information Architecture), OWASP Top 10, AHP

ARTICLE INFO: Received 15 September 2017, Revised 12 October 2017, Accepted 13 October 2017.

[*]Corresponding author is with school of business administration, Chonnam National University, 77, Yongbong-ro, Buk-gu, Gwangju, 61186, KOREA.

E-mail address: s-lee@chonnam.ac.kr

1. 서 론

초창기 웹이 인터넷에 연결된 몇몇 컴퓨터 간에 단순하게 작성한 정보를 공유하는 수준에서 벗어나, 현재는 헤아릴 수 없는 다양한 기기들이 연결되어 정보를 주고받고 있으며 점점 더 그 사용이 늘어가고 있다. 웹이 발전함에 따라 다양한 웹 취약점이 발견되고 이를 악용한 공격이 증가하고 있다. 미국의 애플리케이션 보안서비스 제공업체인 Cenzic社의 연구결과에 의하면 연구를 위해 테스트에 사용된 애플리케이션들의 96퍼센트 가량이 평균 14개의 보안취약성을 지니고 있는 것으로 나타났다고 한다[1]. 2016년 한국인터넷진흥원(KISA)에서 1,242개 웹 사이트에 대해 웹 취약점 점검결과 약73%(895개 사이트)가 취약한 것으로 확인되었다[2]. 이러한 웹 취약점을 해결하고 방어하기 위해 수많은 기업과 기관에서 막대한 비용과 노력을 들이고 있으나 여전히 많은 웹 사이트에서 다양한 웹 취약점이 발견되고 있으며 이를 악용한 공격에 의한 피해가 언론에 자주 보도되고 있다[3].

웹 취약점 점검은 점검 시 대상 시스템의 동작 여부에 따라 정적과 동적 점검으로, 대상 시스템의 내부 구조(예: 소스코드 등)에 대한 인지여부에 따라 화이트박스와 블랙박스 점검으로, 자동화 도구 사용여부에 따라 자동과 수동 점검으로 구분한다[4]. 웹 취약점이 입력 값의 조작에 의해 일어남에 따라 동적 점검이 필수로 요구되고, 대규모 소스코드를 실제 점검하는 것이 한계가 있으므로 블랙박스 방식이 선호된다. 또한 자동화 점검 시 오탐(false positive)이나 서버 부하 문제 등으로 수동점검이 주로 사용되고 있다[5-7]. 즉 현재의 일반적인 웹 취약점 점검은 블랙박스 기반에 동적으로 이루어지는 수동 점검이라 할 수 있다. 이러한 블랙박스 기반에 동적으로 이루어지는 수동 웹 취약점 점검(이하, 현행방식이라 함)은 오탐(false positive), 미탐(false negative) 및 점검 대상 누락 등에 의한 점검 결과 불투명성과 점검자에 따른 점검결과 편향 발생 등과 같은 다양한 문제가 존재한다[8-12].

이에 본 논문에서는 정보구조설계(IA)를 이용하여 점검 대상 웹 사이트를 구조화한 후, 기능 별 점검 우선순위를 부여하고 사이트 특성에 따라 점검 대상 페이지, 기간, 항목 등을 선정할 수 있는 객관적 기준을 제시함으로써, 점검자에 따른 점검결과의 편향을 제거함은 물론이고, 오탐과 미탐 및 점검 대상의 누락 등과 같은 점검 결과의 불투명성을 제거하여 보다 안전한 웹 환경을 만들고자한다.

본 논문의 구성은 다음과 같다. 2장에서는 웹 취약점과 관련된 연구를 통하여 국내·외 웹 취약점 점검의 기준, 웹 취약점 점검방법 및 웹 취약점 점검 절차를 소개하고, 현행 웹 취약점 점검 방법 및 절차의 문제에 대해서 알아본다. 3장에서는 2장에서 알아본 웹 취약점 점검 방법 및 절차와 관련된 문제점을 해결하기 위한 새로운 웹 취약점 점검 방법 및 절차(이하, 제안방식이라 함)를 주요정보통신기반시설 웹 취약점 점검 항목을 기준으로 제시한다. 4장에서는 제안방식에 대한 검증을 위해 현재 운영 중인 웹 사이트를 대상으로 현행방식과 제안방식으로 비교점검을 실시하고, 관련 분야 종사자를 대상으로 한 설문을 통하여 제안방식의 효과성을 입증한다. 마지막으로 5장에서 결론 및 향후 연구 방향에 대해 기술하는 것으로 마감하려 한다.

2. 관련 연구

2.1 웹 취약점 점검

보안 취약점(Security Vulnerability)은 시스템 또는 프로그램에 내재되어 있는 버그(잘못된 부분)로

서 해커는 이를 악용해 시스템에 침입하여 정보 유출, 시스템 파괴 등 유발한다[13]. 취약점 점검은 시스템 또는 프로그램에 내재되어 있는 이런 보안 취약점을 찾는 것을 말한다.

주요정보통신기반시설 취약점 분석 평가 기준에서 제시하는 취약점 분석 평가는 크게 관리적, 물리적, 기술적으로 구분하고, 기술적 취약점 점검은 다시 서버(Window, Unix), 네트워크장비, 보안장비, DBMS, PC, 웹(웹 애플리케이션) 등으로 구분한다[14]. 이중 웹 취약점 점검은 Client에서의 요청에 대한 웹 서버 및 웹 애플리케이션의 동작(반응)을 점검하지만, 다른 시스템(서버, 네트워크장비 등) 점검은 대상 시스템의 설정 값(Config 값)을 점검하는 방식으로 그 특성이 다르며 다음 <표 1>과 같다.

표 1. 기술적 취약점 점검 특성 비교
Table 1. Comparison of Technical Vulnerability Checking Characteristics

구분			점검대상[15]	점검방식		
				자동 도구	Script 파일	수동 (체크리스트)
시스템	서버	윈도우	Config 값	O	◎	△
		Linux	Config 값	O	◎	△
	NW장비		Config 값	O	◎	△
	보안장비		Config 값	△	△	O
	DBMS		Config 값	O	◎	△
	PC		Config 값	O	◎	△
애플리케이션(웹)			반응 값	△	X	◎

※ 점검방식 : ◎(가장 선호), ○(선호), △(보통), X(선호 않음)

2.2 웹 취약점 점검 기준 및 항목

웹 취약점 점검을 위한 점검 항목 및 점검 방법 등에 대해서는 국내·외 다양한 기관에서 기준을 제시하고 있으며, 주요 기준은 다음 <표2>와 같다[16].

표 2. 주요 웹 취약점 점검 기준
Table 2. Major Web Vulnerability Inspection Standard

구분	발행 기관	발행 년도	항목수	특징[16]
OWASP Top 10 2017(RC) [17]	OWASP	2017	상위10개	웹 애플리케이션 취약점 중에서 빈도가 많이 발생하고, 보안상 영향을 크게 줄 수 있는 것들 10가지를 선정하여 2004년부터 3년 주기로 발표
CWE/SANS Top 25 [18]	SANS와 MITRE	2011	25개	웹 애플리케이션을 포함하여 전반적인 소프트웨어의 위험한 오류들을 순서화하여 제시. 보안 코딩 이외의 요소로 인한 웹 보안의 취약점이 주류를 이룸
국정원 홈페이지 8대 취약점 [19]	NCSC	2005	8개	2005년 이후 갱신 안됨. TechNote, Zeroboard 취약점 등 특정 환경 및 버전 이하에서만 발견되는 취약점 존재
홈페이지 SW(웹) 개발보안 가이드 [20]	행정안전부	2012	25개	취약점별로 위험도(상중하 3단계) 구분함. 2012년 이후 갱신안되고 있고, 웹 취약점이 아니라 보안약점에 해당하는 항목(예: 취약한 암호화 알고리즘 사용) 존재
주요정보통신기반시설 취약점 분석·평가 기준(웹) [21]	미래창조과학부	2013	28개	점검 항목이 법적 기준으로 제시(관련고시)되고 있음. 2013년 이후 갱신안되고 있으며 취약점별로 위험도를 구분하고 있으나 모두 '상'으로 분류하고 있음
홈페이지 취약점 진단·제거 가이드 [22]	KISA	2013	21개	기존의 '홈페이지 SW(웹) 개발보안 가이드'와 '주요정보통신기반시설 취약점 분석·평가 기준' 항목을 바탕으로 항목 재 정의함

2.3 웹 취약점 점검 방법 및 절차

웹 취약점 점검은 클라이언트(Client)에서의 요청에 대한 웹 서버 및 웹 애플리케이션의 동작(반응)을 점검하는 것으로, 점검 특성에 따라 크게 정적인 점검과 동적인 점검, 블랙박스 점검과 화이트박스 점검으로 구분할 수 있다[4][23]. 두 분류의 차이는 전자(정적, 동적 점검으로 구분)는 점검 시 대

상 소프트웨어(응용프로그램 or 애플리케이션)가 동작하느냐 하지 않느냐에 따른 구분이고, 후자(블랙박스, 화이트박스 점검으로 구분)는 점검 시 대상 소프트웨어(응용프로그램 or 애플리케이션)의 내부 구조 즉, 소스코드를 포함한 동작 구조를 알고 점검하느냐 모르고 점검하느냐에 따른 구분이다[8].

웹 취약점 점검 방식 분류 및 그 특성은 다음 <표3>과 같으며, 각각의 점검은 점검 주체가 자동화된 도구(툴)이냐 사람이냐에 따라 자동 점검과 수동 점검으로 추가로 구분할 수 있다. 참고로 수동 점검의 경우 시간과 비용의 한계로 점검 대상을 일부 추출(샘플링)하여 점검한다[9].

표 3. 점검 방식 분류 및 특성
Table 3. Classification and characteristics of inspection method

구분		소소코드	실행(동작)	상세 설명
블랙박스	정적	X	X	소스코드 및 프로그램의 실행이 없이 점검 개발명세서의 완결성, 정확성, 일관성 등을 확인하는 점검 방법
	동적	X	O	소스코드 없는 상태에서, 프로그램실행하며 점검 테스트케이스를 만들어 반응값 확인
화이트박스	정적	O	X	소스코드만 보고, 별도 프로그램 실행 없이 점검 일반적으로 도구 등을 사용해 프로그램 상의 치명적인 오류들을 검출
	동적	O	O	소스코드를 가진 상태에서 프로그램을 실행하며 점검 디버깅, 유닛 테스트 및 스크립트를 통한 자동실행 테스트 등

현재 일반적인 웹 취약점 점검은 입력 값의 조작에 의해 일어남에 따라 동적 점검이 필수로 요구된다. 또한 대규모 소스코드를 실제 점검하는 것이 한계가 있으므로 블랙박스 방식이 선호되며, 자동화 점검이 속도 면에서 유리하나 오탐이나 서버 부하 문제가 있어 수동점검이 주로 사용되고 있다[5-7]. 다음 <표4>는 일반적인 웹 취약점 점검의 절차이다.

표 4. 일반적인 웹 취약점 점검 절차[9]
Table 4. Procedure for general Web vulnerability inspection

단계			설명
1단계	준비단계		점검 대상 사이트에 대한 기본적 정보 수집, 점검 기간, 점검 방법, 점검도구 등을 정의
2단계	점검단계	페이지 식별	웹 애플리케이션의 일부인 모든 페이지를 식별(추후, 점검 대상이 됨) ※ 페이지 식별은 자동(웹 크롤러 사용), 수동으로 (프록시로 기록), 반자동(크롤러가 운영자 지원을 요청)의 방식으로 수행할 수 있음
		점검 대상 추출	1단계에서 식별한 페이지에서 점검 포인터(예: 데이터 입력 지점 등)를 식별
		점검 시뮬레이션 및 점검	2단계에서 식별한 점검 포인터에 대해서 어떤 식의 점검을 할지에 대한 판단하고, 점검 진행하여 결과 값(HTTP Response)을 수집
3단계	결과 분석		3단계에서 수집된 결과 값(HTTP Response)대해서 분석하여 취약점을 판단

2.4 현행 웹 취약점 점검의 문제점

앞서 설명한 것처럼 현행방식은 블랙박스 기반에 동적으로 이루어지는 수동 점검으로 웹 사이트 내 일부 페이지(화면)만을 선택(샘플링)하여 점검을 한다. 웹 취약점 점검 방식 및 절차와 관련하여 분야 전문가들이 다양한 연구를 통하여 문제점을 제시하고 있다.

Liu Tie-ming는 2016년 연구에서 동적 웹 취약점 점검의 경우, 코드의 가능한 모든 실행 경로를 탐색 할 수 없기 때문에 코드에 대한 불완전한 분석 및 탐지 문제가 발생한다고 한다[8].

Andrey Petukhov의 2008년 연구에서는 웹 취약점 점검과 관련하여 입력 값을 조작하여 점검하는 웹 취약점 점검 방식(정적, 동적 모두 포함)의 경우

웹 애플리케이션에 도입된 특수 문자 무효처리 로직 등에 의해 실제 존재하는 취약점이 발견되지 않거나, 조건 분기 처리되는 로직의 경우 점검이 누락되는 경우가 발생한다고 한다. 또한, Andrey Petukhov는 모의해킹(penetration testing)과 같은 블랙박스 테스트는 웹 애플리케이션에서 데이터가 어떤 경로로 처리되는지 알지 못하고 단순히 HTTP 응답 값만 보고 판단해야하는 문제도 있다고 한다[9].

Amir Ghahrai의 2016년 연구에서는 동적 점검이 소스코드에 대한 모든 점검이 이루어졌음을 보장하지 못하고 오탐과 미탐의 문제가 있다고 한다[10]. 또한 김세진의 2011년 연구에서는 동적 분석 방법은 애플리케이션 내 특정 조건 경로가 있는 경우 점검이 어려우므로 실제 취약점이 발견되지 않는 미탐(false negative) 문제가 많다고 한다[11].

Dalalana Bertoglio의 2017년 연구에서 보안테스트와 관련된 1145개 논문을 수집하여 그중 모의해킹(penetration testing)과 관련된 1090개 논문을 분석한 결과, 점검 시나리오, 점검방법론, 점검 도구 및 신규 취약점에 대한 점검 등과 관련된 내용의 개선과 연구가 필요하다고 하며, 그 대표적인 예로 점검자가 점검 수행 시 점검의 편향을 피하기 위한 표준화된 절차나 방법이 필요하다고 한다[12].

이상의 웹 취약점과 관련된 여러 연구결과들을 종합한 결과, 현행 웹 취약점 점검 방식 및 절차의 문제는 크게 다음의 두 가지로 요약할 수 있다.

- 점검 결과에 대한 투명성 및 신뢰성 확보 안됨(오탐·미탐·점검 누락 발생 및 어떤 페이지를 어떻게 점검 했는지 등 확인 안됨)
- 점검 시 점검자에 의한 편향 해결위한 방안(표준화된 절차나 방법) 없음

이러한 문제를 해결하기 위해 본 논문에서는 정보구조설계(IA)를 응용하여 점검 대상 웹 사이트의 개별 페이지 식별, 점검 필요성, 점검 우선순위 부여, 점검 대상 표시 등을 통하여 미탐이나 점검 누락 등의 발생을 막아 점검 결과에 대한 투명성 및 신뢰성을 확보한다. 또한, 점검 시 점검자의 경험이나 경력에 따른 편향을 해결하기 위해, 웹 사이트 특성을 고려하여 사이트 특성 별 점검 페이지 선정 기준 및 점검 기간 산정을 위한 객관적 기준을 제시한다.

이 과정에서 본 논문은 개별 페이지에서 서버로 전송되는 파라메타수를 기반으로 동작 크기(Size of Action, SoA)값을 산정하여 활용한다. 별도로 본 논문에서 제안하는 방식에서의 취약점 항목에 대한 기준은 현재 국내에서 법적으로 점검 기준 항목이 명확하게 제시되어있는 주요정보통신기반시설 웹 취약점 점검 항목으로 한다.

3. 웹 취약점 점검 방법 및 절차 개선

3.1 제안방식의 특징

본 논문에서 제안한 웹 취약점 점검 방법 및 절차(이하, 제안방식)의 주요 특징을 요약하면 크게 다음 4가지로 요약할 수 있다.

첫째, 웹 사이트 구조화 및 동작 기능 분석

정보구조설계(Information Architecture, IA)를 이용해서 대상 웹 사이트를 메뉴구조로 목록화한 후 개별 메뉴에 해당하는 페이지(화면, URL)에 서버로의 전송방식(Get/Post), 전송 인자(parameter, 파라메타)를 식별한다[24]. 이때, 조건 분기 처리되는 로직에서 식별 누락이 발생[9]하지 않도록 조건 분기 페이지도 식별(분기조건 명기)하여 기록한다.

개별 페이지가 어떤 동작을 하는 지에 대해 기

능 분석(Functional Specification)을 통해 동작 기능을 식별한다. 동작 기능은 해당 페이지가 수행하는 기능에 따라 7가지(로그인, 조회, 확인, 등록, 변경, 삭제, 이동)로 구분하고 동작 기능별로 점검 우선순위가 부여된다. 동작 기능별 우선순위는 관련 전문가 10인을 대상으로 AHP(Analytic Hierarchy Process)기법을 활용하여 항목 간 쌍대비교 설문을 통하여 선정하였다[25-26].

다음 <표5>는 설문의 일부이고, <그림 1>은 설문 결과를 분석한 내용이다. 설문 결과를 분석한 결과 '로그인 > 조회 > 변경 > 삭제 > 등록 > 확인 > 이동' 의 순서로 동작기능 우선순위가 도출되었다.

표 5. 동작 기능 우선 순위 비교 설문(일부)
Table 5. survey of priority of operational functions(part)

평가항목	매우우선	약간우선	같다	약간우선	매우우선	평가항목
로그인 기능	③	②	①	②	③	조회 기능
	③	②	①	②	③	변경 기능
	③	②	①	②	③	등록 기능
	③	②	①	②	③	삭제 기능
	③	②	①	②	③	확인 기능
	③	②	①	②	③	이동 기능

구분	김A	김B	이C	이D	박E	공F	권G	박H	차I	최J	평균	순위
로그인 기능	0.295	0.283	0.281	0.269	0.177	0.191	0.186	0.177	0.199	0.201	0.231	1
조회 기능	0.210	0.194	0.193	0.196	0.217	0.199	0.199	0.190	0.195	0.137	0.193	2
변경 기능	0.130	0.148	0.147	0.148	0.169	0.170	0.168	0.154	0.160	0.202	0.160	3
등록 기능	0.082	0.109	0.107	0.107	0.123	0.124	0.137	0.142	0.138	0.154	0.120	5
삭제 기능	0.124	0.125	0.131	0.133	0.152	0.152	0.163	0.166	0.162	0.154	0.144	4
확인 기능	0.087	0.087	0.092	0.098	0.113	0.114	0.099	0.102	0.094	0.099	0.098	6
화면 이동 기능	0.072	0.055	0.049	0.049	0.050	0.050	0.049	0.069	0.052	0.054	0.053	7
CI값(일관성계수)	0.0516	0.0447	0.0512	0.0536	0.0549	0.0576	0.0720	0.1047	0.0799	0.0452	-	-
채택여부	O	O	O	O	O	O	O	X	O	O	-	-

그림 1. 쌍대 비교 설문 결과
Figure 1. Results of the questionnaire survey

둘째, 페이지 별 동작 크기(Size of Action, SoA) 값 계산

구조화 및 동작 기능 분석된 웹 사이트의 규모를 정량적인 값으로 산출하기 위하여, 개별 페이지에 단위로 동작 크기(Size of Action, 이하 SoA)값을 구한다. SoA값은 웹 사이트 내 개별 페이지가 점검을 할 수 있는 페이지인가를 식별함과 동시에 웹 사이트의 규모를 정량화시킴으로써 웹 취약점 점검에 필요한 공수 및 기간 산정 등을 객관적할 수 있도록 하기 위해 본 논문에서 도입한 용어이다. SoA값은 클라이언트에서 서버로 전송되는 파라메타의 개수를 바탕으로 파라메타 수가 1개 이상인 페이지가 대상이 되며, 사이트 전체 SoA값 산출식은 다음과 같다.

$$\Sigma SoA(i) = Pn(i) / MPn \quad (단위: 일, day)$$

- SoA(i) : i번째 페이지 동작 크기 값
- Pn(i) : i번째 페이지 전송 파라메타 개수
- MPn : 1 동작크기의 최대 파라메타 수
- i : 페이지 순번으로 1씩 증가

보다 간편한 계산을 위해 위 식에서 1 SoA의 최대 파라메타 수(MPn)의 값을 10으로 정의한다. 따라서 페이지별 SoA값은 파라메타 수를 10개 단위로 끊어서 계산할 수 있고, 파라메타 수가 1~10개까지는 1 SoA값, 11~20개는 2 SoA값, 21~30개는 3 SoA값이 부여된다. 예를 들어, 파라메타 수가 1개인 우편번호 조회 기능을 하는 페이지와 파라메타 수가 10개인 회원등록 기능을 하는 페이지가 있다면 SoA값은 1로서 동일하다.

셋째, 1일 점검 가능 표준 SoA값 계산

웹 취약점 점검은 개별 페이지에 대해 취약점 항목의 존재여부를 체크하는 방식이다. 따라서 점검 기준이 되는 취약점 항목 수에 따라서 점검 시간에 차이가 발생한다. 본 논문에서는 객관적인 점

검 기간 산정을 위해 점검 기준 취약점 항목 수에 따른 1일 점검 가능한 SoA값을 산정하는 기준을 제시한다. 본 논문에서는 주요정보통신기반시설 웹 취약점 점검 기준 항목을 이용하여 표준 1일 SoA 값을 구한다. 표준 1일 SoA값은 1 SoA값을 점검에서 사용할 기준 웹 취약점 점검 항목 수(SVn)으로 점검을 할 때 소요되는 시간을 기준으로 1일(8시간)에 점검할 수 있는 SoA값을 의미한다. 그 산출식은 다음과 같다.

$$SSoA = Td / ((APn * SVn * Tm) / H)$$

- SSoA : 표준 1일 점검 가능 SoA값
- Td : 1일 점검 시간으로 8시간
- APn : 1 SoA의 평균 파라메타 수 (MPn의 1/2인 5를 적용)
- SVn : 기준 웹 취약점 점검 항목 수
- Tm : 파라메타 1개에 대해 1개 웹 취약점 항목 점검에 소요되는 평균 시간(분)
- H : 분을 시간 환산을 위한 변수로 60

앞의 SSoA값 산출 식에서 APn에 5를 적용(MPn을 10이라 했을 때 그 값의 50%로 가정)하고, Tm은 평균 1분이 소요하는 것으로 하여 1을 적용한 후 SSoA값을 계산하면 SSoA=8/((5*28*1)/60)=약 3.4가 된다. 소수점 이하를 올려서 4를 SSoA값으로 사용한다.

3.2 제안방식의 점검 절차(흐름도)

본 논문의 제안방식은 현행방식의 점검 절차인 기본정보수집 ⇨ 점검실시 ⇨ 분석 및 결과보고의 3단계에서, 기본정보수집과 점검실시 사이 단계에 사이트 구조화 및 기능 분류 ⇨ 사이트 전체 SoA값 산정 및 사이트 전체 점검 기간 산정 ⇨ 점검 대상 페이지 확정의 3단계가 추가된 6단계이다. 다음 <그림2>는 개선방식의 점검절차로서 붉은 박스 부분이 추가된 절차이다.

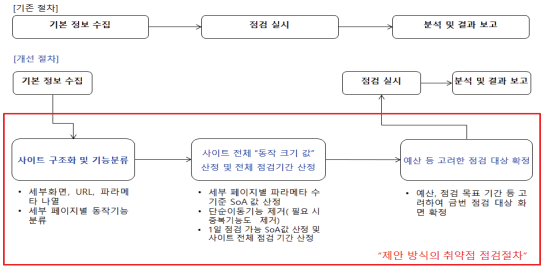


그림 2. 제안방식 점검 절차
Figure 2. Procedure for Proposal method

3.3 제안방식의 단계별 수행 내용

본 절에서는 제안방식에서 새롭게 추가된 3단계에서 수행할 내용에 대해서 설명한다.

가. 웹 사이트 구조화 및 기능 분류 단계

점검 대상 웹 사이트를 정보구조설계로 세부 메뉴단위로 세분화하여 페이지(화면, URL), 전송방식, 전송 파라메타를 식별하고, 세부 페이지를 7가지 동작 기능 중 어디에 해당하는 지 구분하는 단계이다. 다음 <표 6>은 그 결과에 대한 예시이다.

표 6. 웹 사이트 구조화 및 기능 분류 결과(일부)
Table 6. Classification result of website structure and function(part)

메뉴			페이지 URL	전송 방식	파라메타		동작 기능
1depth	2depth	3depth			수	값	
자료실	-	-	http://192.168.0.1/docu/info.html	Get	0	-	단순 이동
자료실	검색	-	http://192.168.0.1/docu/search.jsp	Post	3	keyword=&page=1&	조회
로그인	확인	-	https://192.168.0.1/Member/Login	Post	2	onln_id=&custpw=	로그인
로그인	확인	(자동 전송)	http://192.168.0.1/Member/login_ok	Post	4	onln_id=&custpw=&referrer=l&returnUrl=	로그인
~~생략~~							

나. 웹 사이트 전체 SoA값 및 점검기간 산정 단계

개별 페이지의 SoA(i)값을 구하여 웹 사이트 전체의 SoA값(∑SoA(i))을 산정하고, 그 값을 바탕으로 전체 점검기간을 산정하는 단계이다. 개별 페이지의 SoA(i)값이 0인 페이지는 점검대상에서 제외된다. 다음 <표 7>은 그 결과에 대한 예시이다.

표 7. 페이지별 SoA값 산정 결과(일부)
Table 7. Result of calculation of SoA value per page(part)

메뉴			페이지 URL	전송 방식	파라메타		동작 기능	SoA 값	점검 대상
1depth	2depth	3depth			수	값			
자료실	-	-	http://192.168.0.1/docu/info.html	Get	0	-	단순 이동	0	X
자료실	검색	-	http://192.168.0.1/docu/search.jsp	Post	3	keyword=&page=1&	조회	1	O
로그인	확인	-	https://192.168.0.1/Member/Login	Post	2	onln_id=&custpw=	로그인	1	O
로그인	확인	(자동 전송)	http://192.168.0.1/Member/login_ok	Post	4	onln_id=&custpw=&referrer=l&returnUrl=	로그인	1	O
~~생략~~									

웹 취약점 점검에서 사용할 기준 취약점 점검 항목(SVn)으로 본 논문에서는 주요정보통신기반시설 웹 취약점 항목을 사용하며, 이때 1일 점검 가능 동작크기 값(SSoA값)은 4가 된다. 웹 사이트 전체의 SoA값(∑SoA(i))을 SSoA값으로 나누면 웹 사이트 점검기간이 된다.

다. 점검 대상 페이지 선정 및 확정 단계

웹 사이트 내 최종 점검 대상 페이지를 선발·확정하는 단계로 점검 시점의 예산, 일정, 인력 등의 이유로 산정된 점검기간 보다 기간이 줄어드는 경우 동작 기능 우선순위(로그인 > 조회 > 변경 > 삭제 > 등록 > 확인 > 이동)에 따라 점검 대상 페이지를 선정한다.

4. 제안방식 적용 및 평가

본 논문에서는 제안방식의 효과성을 평가하기 위해 실제 웹 사이트를 대상으로 현행방식과 제안방식으로 점검한 결과를 비교하고, 관련 분야 인력을 대상으로 설문조사를 실시한다.

4.1 웹 취약점 점검 및 결과

가. 취약점 점검 사이트 정보 및 진행 방식

취약점 점검은 온라인 서비스를 제공하는 웹 사이트를 대상으로 6명의 인원이 2개조(조별 고급, 중급, 초급 경력 각 1명으로 조별 3명)로 구성하였고, 각 조의 구성들은 서로 독립적으로 개별 점검하였다. 다만, 조별 고급 인력은 점검 조장역할을 하였으나 점검 개시·종료 및 일일 점검 결과를 취합 등의 역할을 수행하였다. 동일 조 내의 인력에 대해 별도의 역할 구분 없이 개별 점검을 수행한 이유는 점검인력의 경력차이가 현행방식과 비교 시 제안방식에서 어떤 차이가 발생하는지 알고자 함이다.

A조는 현행방식으로 B조는 제안방식으로 점검하였으며, 점검 기간은 조별 10일씩(A조 : 2017.5.22.~6.2, B조 : 6.5~6.20) 온라인으로 점검을 실시하였다. 현행방식으로 점검하는 A조가 제안방식을 모방하지 못하도록, 조별 점검 기간을 다르게 하고 A조를 B조보다 먼저 점검하도록 하였다. 또한, 점검 시 동일 조원끼리 발견 취약점 정보 공유 등을 방지하기 위해 조원 간 의견 교환을 금지했다. 별도로, 제안방식으로 점검하는 B조의 경우 점검 대상 페이지 각각에 대해 취약점 점검 항목을 모두 체크하는 방식으로 점검하도록 요청하였다. 다음 <표 8>은 점검 사이트 관련 정보이다.

표 8. 웹 사이트 정보
Table 8. Website information

구분	상세 정보
Web서버	IBM X3550, Win2012 R2
WAS서버	IIS, ASP
DB서버	IBM X3550, Win2012 R2
DBMS	MS SQL 2012
서비스	온라인 쇼핑몰 (B2C 서비스)
페이지 수	약 100페이지 이상 (일부 중복페이지 제외)
특기 사항	2015년 및 2016년도 대외인증 취득 준비 시 웹 취약점 점검 실시(본 사이트 포함 7개 사이트를 중급 1인이 10일간 웹 취약점 점검 실시) - 2015년도 경우 총2개(정보누출, XSS 각 1개) 발견 및 조치 완료 - 2016년도 경우 총2개(XSS, 데이터 평문 전송) 각 1개 발견 및 조치 완료

나. 취약점 점검 실시

제안방식을 통해 사이트를 구조화한 결과, 총 108개의 페이지가 식별되었고 타 사이트로 이동하는 경우를 제외한 결과, 총 100개가 웹 사이트 내 페이지로 확인되었다. 100개 페이지를 대상으로 기능 분류 및 페이지별 SoA값을 계산한 결과, 67 SoA값이 도출되었다.

표 9. 웹 사이트 구조화 및 SoA값 산정 결과
Table 9. Classification result of website structure and calculation of SoA value

구분	페이지	SoA값			제외 이유
		전체	제외	대상	
로그인	4	4	0	4	-
조회	37	28	16	12	타 페이지와 동일
변경	4	4	0	4	-
삭제	3	3	1	2	최종 삭제 처리
등록	8	8	1	7	최종 등록 처리
확인	8	8	5	3	최종 결제 처리
이동	36	12	12	0	단순 페이지 이동
합계	100	67	35	32	-

타 페이지와 동일한 기능을 하거나, 최종 삭제·등록 등으로 DB변경이 크게 발생하는 기능이거나, 단순 페이지 이동인 경우 등을 제외한 결과 총 32 SoA값이 점검 우선 페이지로 확인되었다. 최종 삭제·등록 등의 처리는 사이트 안전성 고려하여 본 논문의 점검에서는 제외하였다. 다음 <표 9>는 웹 사이트 동작기능 분류 및 SoA값 산정 결과이다.

SSoA(표준 1일 동작 크기 값)이 4이므로, 점검 대상 SoA값 32를 4로 나누면 총 8일이 B조(제안방식 적용 조)의 표준 점검 기간이 된다. 본 논문에서는 점검자별 총 10일간의 점검을 사용하므로, <표 9>에서 타 페이지와 동일 기능으로 제외되었던 조회기능의 12 SoA를 추가로 점검하며, 제안 방식으로 점검하는 B조의 일자별 점검 대상 동작 기능은 다음 <표 10>과 같다.

표 10. 일자별 동작 기능 및 동작 기능 값(B조)
Table 10. Function and values by date(Team B)

구분	일자									
	1일	2일	3일	4일	5일	6일	7일	8일	9일	10일
동작 기능	로	조	조	조	변	삭, 등	등	등, 확	조	조
SoA 값	4	4	4	4	4	4	4	4	6	6

※로(로그인), 조(조회), 변(변경), 삭(삭제), 등(등록), 확(확인)

다. 취약점 점검 결과

취약점 점검 결과 크게 발견된 웹 취약점의 종류와 개수로 구분하여 집계하였다. 발견된 웹 취약점의 종류는 웹 사이트 전체가 기준이며 사이트 내 웹 취약점의 다양성을 의미(점검 기준 취약점 항목인 28개가 최대)하며, 공격자 입장에서는 다양한 방법으로 공격이 가능함을 의미한다. 이에 반해 발견된 웹 취약점 개수는 웹 사이트 전체를 기준으로 취약점 종류와 상관없이 얼마나 많은 취약점이 존재하는 가를 보는 것으로, 개수가 많다는 것

은 취약점이 있는 페이지 및 파라메타가 많다는 의미로 공격자 입장에서는 공격을 할 수 있는 포인터가 많다는 의미가 된다. 다음 <표 11>, <그림 3>, <표 12>, <그림 4>는 점검방식별 점검 결과를 취약점 종류 및 개수로 각각 비교한 것이다.

표 11. 발견 취약점 종류 비교
Table 11. Comparison of types of discovery vulnerabilities

취약점종류		1일	2일	3일	4일	5일	6일	7일	8일	9일	10일	전체
A조 (현행 방식)	초급	2	4	5	6	7	7	7	8	8	8	8
	중급	3	5	6	8	9	10	10	10	10	10	10
	고급	4	7	9	10	10	10	10	10	10	10	10
B조 (제안 방식)	초급	4	6	6	6	10	10	10	10	10	10	10
	중급	4	6	6	6	10	10	10	10	10	10	10
	고급	4	6	6	6	10	10	10	10	10	10	10

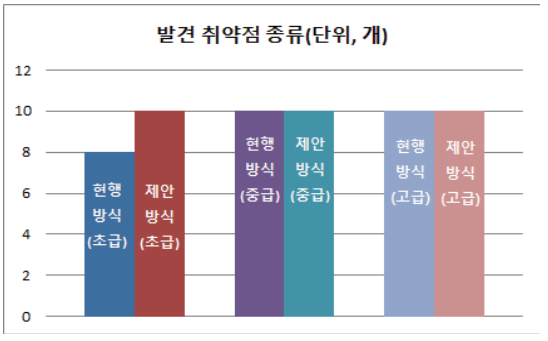


그림 3. 발견 취약점 종류 비교(총계)
Figure 3. Comparison of types of discovery vulnerabilities(Total)

표 12. 발견 취약점 개수 비교
Table 12. Comparison of the number of discovery vulnerabilities

취약점종류		1일	2일	3일	4일	5일	6일	7일	8일	9일	10일	전체
A조 (현행 방식)	초급	2	4	5	6	7	7	7	9	10	11	11
	중급	3	6	7	10	11	13	14	16	18	22	22
	고급	4	8	10	14	17	19	20	22	24	27	27
B조 (제안 방식)	초급	4	9	13	19	26	28	31	36	50	64	64
	중급	4	14	18	26	33	37	41	49	65	78	78
	고급	4	14	18	26	33	37	41	49	65	78	78

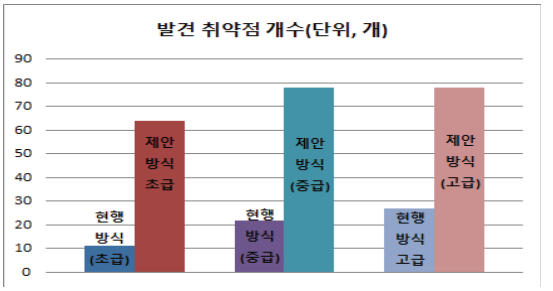


그림 4. 발견 취약점 개수 비교(총계)
Figure 4. Comparison of the number of discovery vulnerabilities(Total)

4.2 설문 조사 및 결과

가. 설문 조사

본 논문의 제안방식은 취약점 점검 계획 수립에서 점검 수행 및 후속조치(취약점 제거 등)까지 취약점 점검관련 전반적인 부분의 개선을 제시하고 있는데, 취약점 점검 결과만으로 유효성을 확인에 일부 어려움(예: 제안방식이 후속조치 관련 유효성 등)에 이 있어 관련 분야 전문가를 대상으로 설문을 통해 취약점 점검 결과와 상호보완 확인한다. 웹 취약점 관련 설문 조사는 Krueger and Casey의 심층인터뷰 분석지침을 참조하여 다음 <표 13>과 같이 총 5단계의 절차로 수행하였다[27].

표 13. 설문 진행 단계
Table 13. Questionnaire progress stage.

구분		상세 활동 및 내용	기간
1단계	설문준비	• 관련 연구 및 설문지 개발	5.1~5.12
2단계	설문 검증 및 수정	• 설문 대상자 3인을 대상으로 설문을 진행하여 설문 오류 등 검증	5.15~5.26
3단계	설문지 배포 및 설문 실시	• 설문지 전달 및 설문배경, 목적 설명 • 설문 실시 후 회수	5.29~6.16
4단계	설문 결과 분석	• 설문 내용 분석 • 설문 결과 오류 응답 제거	6.19~6.30
5단계	요약정리	• 설문 결과 요약 및 피드백	7.3~7.7

검증 과정을 거쳐 완성된 설문은 5개영역에 총 97개 문항으로 구성되어있다. 5개 영역 중 Part5의 영역은 다른 영역과 다르게 웹 취약점 점검과 관련된 기업 또는 기관의 담당자를 대상으로 한 설문이며, 영역별 구성 내용은 <표 14>와 같다.

표 14. 설문 구성 및 문항 수
Table 14. Survey composition and number of items

설문 영역	설문 목적	문항수
설문자 정보	• 설문자 업무 및 웹 관련 경력 조사	8문항
점검 관련 일반사항	• 웹 취약점 점검 수행 목적, 기간, 수행 계획 수립 시 문제점 등 조사 • 점검 계획 수립 시 제안 방식의 우수성 및 필요성 검증	26문항
점검 방식과 형태 관련	• 웹 취약점 점검방법(자동vs수동) 및 점검형태(전수 vs 샘플링)등 조사 • 현행 취약점 점검 방식/형태의 문제점 개선여부 등 조사	17문항
점검 항목 관련	• 웹 취약점 점검 항목, 점검항목의 활용형태 조사 및 제안 방식을 통한 문제점 개선 여부	27문항
관리자 측면	• 웹 취약점 대응 관리 측면에서 제안방식의 문제점 개선 여부 조사	19문항

나. 설문 조사 대상

본 논문에서는 웹 취약점 점검 계획의 수립 및 점검, 발견된 취약점에 대한 조치(제거), 그리고 후속 조치 등과 관련하여 영역별 실무 담당자이 생각하는 현행방식의 문제점 및 제안방식의 유효성 등을 확인하기 위한 설문을 하였다. 이를 위해 총 100명의 설문 인원을 기업 및 기관의 보안관련 인력 및 웹 사이트 운영 인력 35명, 기업 및 기관의 웹 사이트와 관련된 개발 인력 25명, 웹 취약점 점검을 포함한 컨설턴트 40명으로 구성하였다.

설문 응답자는 필수적으로 웹 및 웹 취약점과 관련된 지식이나 업무 경험을 가지고 있는 인원을 그 대상으로 하였다. 설문을 약 3주에 걸쳐 배포하고 회수한 결과 총 100명 중 72명이 응답에 응했으며, 응답자 72명 중 일관성 부족 등으로 무효 처리된 응답자 9명을 제외한 61명이 유효한 응답으로 확인되었다.

4.3 제안방식의 유효성 및 우수성

본 절에서는 앞선 취약점 점검 결과와 설문 조사 결과를 분석하여 본 논문의 제안방식이 현행방식의 문제점을 해결할 수 있음을 증명하고자 한다.

가. 점검 결과 투명성 및 신뢰성 확보

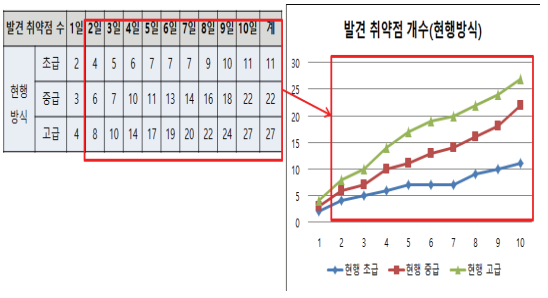
현행방식의 문제점 중 하나인 점검 결과의 불투명성 및 낮은 신뢰성은 실제 취약점 점검 결과 조치를 포함하여 전체적인 담당자 입장에서는 가장 큰 어려움으로 실제 설문응답자의 87.5%가 점검 결과의 불투명성을 가장 큰 어려움으로 응답했다. 또한, 점검 결과의 불투명성으로는 점검한 화면 확인 안되는 것(65%)과 발견된 취약점이 다른 페이지(화면, URL) 등에 추가 존재확인 안되는 것(35%) 등이 있다고 응답했다.

이러한 점검 결과의 불투명성은 어떤 페이지를 어떻게 점검했는지 모르기 때문에 발생하는데, 이는 점검 대상 페이지에 대한 취약점 항목의 점검 방법과 관련이 있는데 점검 대상 페이지와 취약점 항목사이의 점검 방식(체크 방식)을 묻는 질문에 응답자 전원이 세부 페이지(화면, URL)에 대해 취약점 항목을 전수 체크하는 방식으로 점검을 하지 않는다고 응답(취약점 항목 중심으로만 점검(72.1%), 페이지중심으로만 점검(27.9%))하였다.

현행방식의 이런 문제를 해결하기 위해 본 논문에서는 대상 웹 사이트를 구조화하여 개별 페이지를 식별하여 동작기능 구분 및 SoA값을 산정하여 점검 대상 페이지 여부를 표시하였다. 이 과정에서 조건 분기 처리가 있는 경우 각 조건에 따른 페이

지도 식별하여 점검 누락이 발생하지 않도록 하였다. 또한, SoA값과 동작 기능 우선순위를 이용하여 점검 시 어떤 대상 페이지가 점검되었는지를 알 수 있도록 하였고, 점검 대상 페이지 각각에 대해 취약점 점검 항목 전체를 점검하도록 하였다.

<그림 5>는 점검 사이트를 과거 2015년도 2016년도처럼 점검(중급 경력자 1인이 1일 점검)했을 때 2일차이후 취약점(붉은 박스 및 원)은 발견되지 않을 가능성이 높다는 것을 의미한다. 추가로, 현행방식의 경우 점검 시 마다 대상 선정이나 점검 방식 등이 점검자에 따라 다르므로, 현행방식대로 매년 점검을 한다고 해도 2일차, 3일차 순서로 취약점이 발견 제거된다는 것을 보장하지는 못한다.



발견 취약점 수		1일	2일	3일	4일	5일	6일	7일	8일	9일	10일	계
현행 방식	초급	2	4	5	6	7	7	7	9	10	11	11
	중급	3	6	7	10	11	13	14	16	18	22	22
	고급	4	8	10	14	17	19	20	22	24	27	27

그림 5. 이전년도와 동일하게 점검 시 잔존 취약점
Figure 5. As with previous years, residual vulnerability at inspection

특히, <그림 6>에서 보는 바와 같이 현행방식의 경우 점검 인력 경력에 따른 점검 결과의 편차(취약점 개수 기준, 초급 vs 고급 약 100%차이)가 크며 이는 점검자에 따라 발견되지 않을 취약점이 더 존재할 수도 있다는 의미이다. 이에 비해 제안방식인 <그림 7>은 경력별 편차(취약점 종류는 편차가 없고, 취약점 개수에서 초급 vs 고급이 약 10%미만)적어 점검결과의 차이가 없음을 알 수 있다.

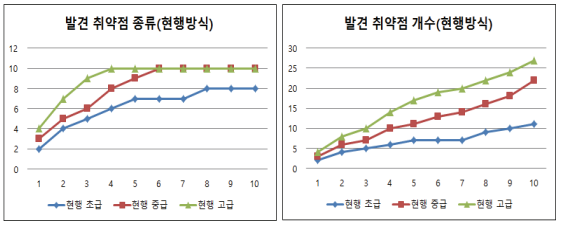


그림 6. 경력 별 점검 결과 편차(현행방식)
Figure 6. Carrier specific inspection result deviation(Current Method)

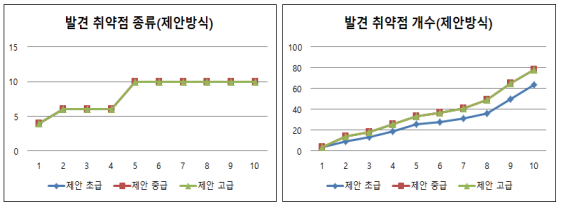


그림 7. 경력 별 점검 결과 편차(제안방식)
Figure 7. Carrier specific inspection result deviation(Proposed Method)

<표 15>와 <그림 8>은 초급 경력자를 기준으로 가장 일반적인 웹 취약점인 XSS(크로스사이트 스크립트)에 대해 현행방식과 제안방식으로 점검 시 발견한 개수를 비교한 것으로, 현행방식의 경우 실제 사이트 내 존재하는 취약점의 10%정도(제안방식을 100이라 했을 때)만 찾아내고 나머지 90%정도(<그림 8>의 화살표 부분)를 찾아내지 못한다는 것을 의미한다.

표 15. 조별 XSS발견 개수(초급 기준)
Table 15. A comparison of the number of XSS by Team

구분		현행방식	제안방식	변화
XSS	발견 페이지 수	4 개	9 개	225%(↑)
	발견 개수	4 개	44 개	1000%(↑)
	페이지 당 평균	1 개	약 4.9 개	약 490%(↑)

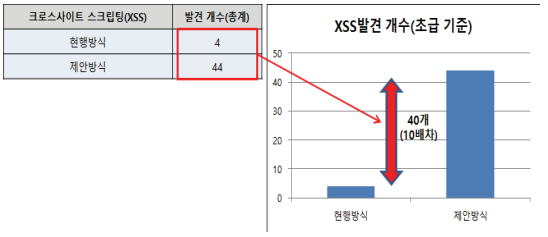


그림 8. 점검방식 별 XSS발견 개수 비교
Figure 8. A comparison of the number of XSS by Team

즉, 제안방식은 현행방식이 찾아내지 못하는 잔존 취약점까지 모두 찾아내고, 점검자 경력에 따른 편차가 거의 없어 점검자에 따른 누락이 발생 가능성이 낮기 때문에 점검 결과의 투명성과 신뢰성 확보가 가능하다는 것을 알 수 있다.

추가적으로 점검한 페이지(화면, URL)가 확인 가능하면 업무에 도움이 될지를 묻는 질문에 응답자 전원이 도움(매우 도움(50%), 조금 도움(50%))이 된다고 답했다. 사이트 내 다른 페이지에 대한 확인 등의 유사 조치에 도움(47%)이 되거나, 점검현황 관리에 도움(27.7%) 또는 다음 점검 시 계획 수립 등에 도움(25.3%)이 된다고 응답해서, 본 논문의 제안방식이 점검 결과의 투명성과 신뢰성확보에 도움이 될 것임을 알 수 있다.

나. 점검자에 의한 편향 해결위한 방안(표준화된 절차, 방법)

현행방식의 경우, 점검 대상 페이지의 선정에서 점검까지를 점검자의 경험에 전적으로 의존하므로 점검자에 따른 편향이 발생할 수밖에 없다. 실제 점검 페이지(화면, URL) 선정 시 기준이 존재하는가를 묻는 질문에 91.8%가 특별한 기준 없이 점검자가 과거 경험 등을 바탕으로 임의 선정한다고 응답했다. 또한, 점검 대상 페이지(화면, URL) 선정을 위한 객관적 기준이 있을 때 활용여부와 어떤 면에서 도움이 될지를 묻는 질문에, 모든 응답자(100%)가 적극 또는 부분 활용을 할 것이며, 응답자의 85.2%가 점검인력의 경력에 따른 차이를 없앨 수 있는 것을 가장 큰 효과로 답해 점검 대상 페이지(화면, URL) 선정을 위한 객관적 기준의 필요성이 아주 높음을 알 수 있다.

본 논문에서는 사이트 구조화, 동작 기능 식별 및 SoA값을 이용하여 점검 대상 페이지의 선택 방법 및 점검 기간의 산정 방법을 수식과 함께 제시하였다. 이를 적용하여 실제 사이트를 대상으로 한 취약점 점검에서 앞의 <표 9>에서 보는 바와 같이 총 108개의 페이지에서 67개의 점검 가능한 페이지가 식별되었고, 동작 기능 분류 등을 통해 최종 점검 대상으로 32개의 페이지가 식별되었다. 식별된 32개 페이지는 앞의 <표 10>에서 보는 바와 같이 동작 기능에 따라 점검 순서도 정의가 되었다. 또한, 기준이 되는 취약점 점검 항목 수에 따라 1일 점검 가능한 SoA값의 산정이 가능하였고, 그를 바탕으로 객관적인 점검기간의 산정이 가능하였다.

<그림 9>와 <그림 10>은 점검 조별로 발견 취약점 종류와 개수를 일자별로 누적한 것으로, 현행방식(그림 좌측)과 비교 시 제안방식(그림 우측)의 경우 취약점 종류나 개수에서 점검자 간의 격차가 현격히 줄어든 것을 확인 할 수 있다. 이는 제안방식이 사이트 구조화, 동작 기능 분류, SoA값 산정, 점검 대상 페이지 선정, 기간산정 등의 과정이 표준화됨에 따라 점검자에 따른 편향을 해소하였음을 알 수 있다.

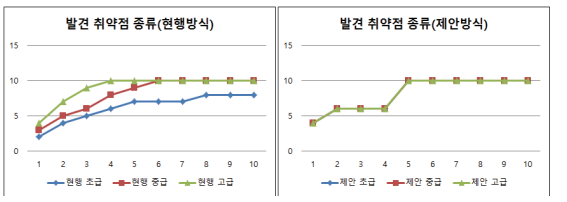


그림 9. 발견 취약점 종류 비교(일자별)
Figure 9. Comparison of types of discovery vulnerabilities(By date)

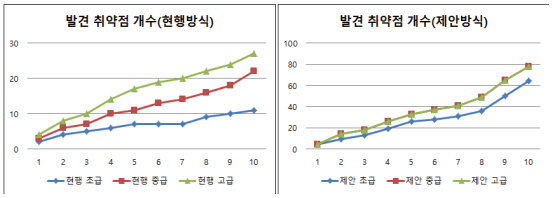


그림 10. 발견 취약점 개수 비교(일자별)
Figure 10. Comparison of the number of discovery vulnerabilities(By data)

5. 결 론

웹이 발전하면 할수록 악의적 공격 등을 통한 정보의 유출이나 서비스 방해 등과 같은 다양한 웹 취약점이 발생하고 이를 악용한 공격이 증가하고 있다. 이를 해결하고 방어하기 위해 수많은 기업과 기관에서 막대한 비용과 노력을 들이고 있으나 여전히 많은 웹 사이트에서 다양한 웹 취약점이 발견되고 있으며 이를 악용한 공격에 의한 피해가 계속 보도되고 있다.

본 논문에서는 이러한 문제의 발생 원인에 대해 현재의 웹 취약점 점검 방법 및 절차를 중심으로 살펴보고 그에 대한 개선을 위해 새로운 취약점 점검 방법 및 절차를 제시하였다. 그리고 실제 사이트를 대상으로 현행방식과 제안방식으로 비교 점검하고, 관련 분야 종사자를 대상으로 설문 조사한 결과를 분석하여 제안방식의 유효성을 입증하였다. 특히, 현행방식과 제안방식의 취약점 비교 점검결과에서 보듯이 제안방식이 현행방식으로는 발견하지 못하는 많은 수의 취약점을 발견하는 것을 알 수 있고 이는 점검기간이 길어질수록 그 효과가 크다는 것을 알 수 있다. 이는 일정기간 이상의 점검기간이 확보가 되지 않으면 어떤 웹 취약점 방법도 그 효과를 충분히 발휘하지 못한다는 의미가 되며, 이는 각 기업의 보안담당자 및 경영진에서 점검 계획 수립 시 고려해야할 요소가 된다.

본 논문의 제안방식은 취약점 점검 계획 수립에서 결과 조치 등을 담당하는 담당자 및 실제 취약점 점검을 실시하는 컨설턴트 모두가 활용가능한 방식이다. 또한 본 논문의 제안방식은 개발단계에서 정적분석에서도 활용가능하며 특히 세부 페이지단위로 관리가 필요한 경우 더 유용하다.

본 논문의 제안방식은 현행방식(동적, 수동점검)의 문제점을 개선하였으나 나날이 발전하는 웹 취약점 관련 위협에 대한 그 근본적 대안은 될 수 가 없다. 따라서 각 점검 방식이 가지는 장점을 잘 파악하고 개별 기업에 적절한 형태의 점검 방식을 적용할 필요가 있다. 특히, 예산 및 기간적 여유가 있는 경우 자동화 도구를 이용한 점검과 제안방식을 병행한다면 그 효과는 배가 될 수 있다. 이를 위해 정적점검에서의 활용방안과 자동점검과의 연계 활용방안 등에 대해서도 추가적인 연구가 필요하다.

또한 세부 페이지별 동작크기(SoA)값을 계산 시 단순히 파라메타의 개수만으로 판단하고 파라메타별 중요도를 고려하지 않았으므로, 파라메타별 중요도를 고려하여 가중치를 적용하여 동작크기(SoA)값을 계산하는 부분에 대한 연구도 필요하다.

References

[1] D. S. Jeong, *Many security vulnerabilities found in most web applications*, http://digitalyeogie.com/entry/49451?locPos=25Q&ts=1487390696&page=4862, Apr. 2017

[2] KISA, *Web vulnerability analysis and technical support,* Korea Internet Security Agency, 2016

[3] M. S. Kim, *Bombu hacking is attacking web vulnerability DB-the reason for the 2 million personal information was revealed,*

http://news.kukinews.com/news/article.html?no=317262, May 2017.

[4] M. Y. Bae, *A study on HTML5 security fragility analysis and inspection technique*, Department of Information Communication Engineering Graduate School Andong National University, pp. 31-38, Jun. 2016.

[5] K. I. Seo, and E. M. Choi, *Dynamic analysis based on AOP for checking security vulnerability*, Journal of KISS : Software and Applications, Vol. 37, No. 10, pp. 773-778 (6 pages), Oct. 2010.

[6] K. G. Kim, *Internet hacking and security-Introduction to Information Security and Practice*, Hanbit academy, 2015.

[7] T. Y. Park, *Study on the improvement of web vulnerability diagnosis of companies using Web Privacy Impact Assessment(WPIA),* Graduate school of Information Communication, Konkuk University, pp. 5-31, Aug. 2015.

[8] T. M. Liu, L. H, Jinag, J. S. Zhu, and G. Meng, *The interactive mechanism of static and dynamic analysis in the reverse analysis of embedded software*, International Journal of Multimedia and Ubiquitous Engineering, Vol. 11, No. 10, pp. 33-44, 2016.

[9] A. D. Petukhov, and D. T. Kozlov, *Detecting security vulnerabilities in web applications using dynamic analysis with penetration testing,* OWASP Application Security Conference, Ghent, Belgium, pp. 19-22, May 2008.

[10] G. H. Amir, *Static analysis vs dynamic analysis in software testing*, https://www.testingexcellence.com/static-analysis-vs-dynamic-analysis-software-testing/, May 2016.

[11] S. Z. JIN, and K. G. DOH, *Detection of DOM-based cross-site scripting with dynamic request*, KOREA INFORMATION SCIENCE SOCIETY, Vol. 38, No. 2C, pp. 168-171, Nov. 2011.

[12] D. D. Bertoglio, and A. F. Zorzo, *Overview and open issues on penetration test,* the Brazilian Computer Society Vol. 23, No. 2, 2017.

[13] TTA, *Dictionary of information and communication terms*, http://terms.tta.or.kr/dictionary/dictionaryView.do, May. 2017.

[14] KISA, *Detailed information on how to analyze and evaluate key information infrastructure vulnerabilities,* Korea Internet Security Agency, 2014.

[15] A3 Security, *Explanation of technical vulnerability diagnosis consulting*, http://www.a3sc.co.kr/01/cons02_6.php, May. 2017.

[16] J. H. Lee, and S. J. Lee, *A study on improvement of web vulnerability inspection standard of the Critical Information and Communication Infrastructure,* Proceedings of the 21st KKITS Spring Conference Vol. 11, No. 1, pp. 76-79, Jun. 2017.

[17] OWASP, *Category:OWASP Top Ten Project,* https://www.owasp.org/, May. 2017.

[18] J. C. Moon, and S. J. Cho, *Vulnerability analysis and threat mitigation for secure Web application development*, Journal of the Korea Society of Computer and Information Vol. 17, No. 2, pp. 127-137, Feb. 2012.

[19] NCSC, *Homepage security management manual*, National Cyber Security Center, 2005

[20] KISA, *Website development for information system development operators SW (web)*

development security guide, Korea Internet Security Agency, 2012.

[21] Korea Law Information Center, *Vulnerability analysis and evaluation criteria for major IT infrastructure facilities(No. 2013-37)*, http://www.law.go.kr/, May. 2017.

[22] KISA, *Homepage for information system development operator guide to vulnerability diagnosis and removal,* Korea Internet Security Agency, Dec. 2013.

[23] Wiki, *Web Vulnerability check,* https://ko.wikipedia.org/wiki/, May 2017.

[24] J. M. Lee(Translation), *Information architecture for world wide web(P. Morville, and L. Rosenfeld),* Hanbit academy, 2011.

[25] R. W. Saaty, *The analytic hierarchy process -what it is and how it is used,* Mathematical Modelling, Vol. 9, Issues 3-5, pp. 161-176, 1987.

[26] M. Alexander, *Decision-making using the analytic hierarchy process(AHP) and SAS/IML*, http://analytics.ncsu.edu/sesug/2012/SD-04.pdf, Jun. 2017.

[27] R. A. Kruger, and M. A. Casey, *Focus groups: A practical guide for applied research 5th edition*, Sage Publication Inc, Thousand Oaks, California, 2015.

웹 사이트 구조화 및 페이지별 동작크기 계산을 통한 동적 웹 취약점 점검 방법 및 절차 개선 연구

이재호[1], 이상준[2]

[1] *전남대학교 정보보안협동과정*

[2] *전남대학교 경영학부*

요 약

웹이 발전함에 따라 다양한 웹 취약점이 발견되고 있다. 많은 기업과 여러 기관에서 웹 취약점을 제거하기 위해 노력하고 있으나 웹 취약점은 줄어들고 있지 않다. 웹 취약점 점검은 그 특성상 동적점검이 필수적으로 요구되며, 정확한 점검을 위해서는 수동점검이 바람직하다. 수동으로 진행하는 동적점검의 경우, 오탐이나 점검누락 그리고 점검자에 따른 점검의 편향(차이) 등과 같은 다양한 문제가 존재한다. 본 논문에서는 오탐이나 점검누락 등을 방지하고 점검자의 편향을 해결할 수 있는 점검방법과 절차를 제시한다. 제안방식은 점검 대상 웹 사이트를 정보구조설계(IA)를 이용하여 구조화한 후, 세부 페이지들을 7가지의 동작 기능으로 분류한다. 세부페이지들은 파라메타 개수를 이용하여 크기를 구하고, 세부페이지의 크기를 합하여 웹 사이트 전체의 크기를 수치화한다. 점검에 사용할 웹 취약점 항목을 이용하여 하루 동안에 점검 가능한 페이지 수를 계산한 후, 그 값을 이용하여 웹 사이트 전체 크기에서 전체 점검 일정을 계산한다. 제안방법과 현재방법으로 취약점 점검하여 발견한 취약점 개수를 비교하고, 관련 분야 종사자를 대상으로 한 설문결과를 분석하여 제안방법의 타당성을 검증 하였다. 제안방식은 동적점검뿐만 아니라 정적점검에서도 응용이 가능하다.

Jae-Ho Lee received the bachelor's degree in the Department of Electrical Engineering from the Chung-Ang University in 1995. He received the M.S. degree in the Graduate School of Information & Technology, Sogang University. His current research interests include ISMS(Information Security Management System), PIMS(Personal Information Management System), Web vulnerability.

E-mail address: haksa26@hanmail.net

Sang-Joon Lee received the B.S., M.S. and Ph.D. degrees in Computer Science and Statistics from Chonnam National University in 1991, 1993 and 1999, respectively. From 1995 to 2006, he was in Seonam University and Shingyeong University as an assistant professor. Since 2007, He has been with Chonnam National University as a professor in the school of business administration. His current research interests include Management Information Systems, Software Engineering, IT Service, Information Security and Ubiquitous Business. He is a life member of the KKITS.

E-mail address: s-lee@chonnam.ac.kr